(51) International Patent Classification [7] :		(11) International Publication Number:	**WO 00/22547**
G06F 17/00	**A2**	(43) International Publication Date:	20 April 2000 (20.04.00)

(54) Title: GENERATION AND MAINTENANCE OF DOCUMENT DATABASES

(57) Abstract

A method for generating or maintaining a plurality of document databases comprising design elements and comprised within one or several applications is described as well as a system for performing the method. The method comprising the step of standardising at least some of the design elements of the document databases by communicating design features to and applying the design features to each of the plurality of document databases from a common library. The common library has a data storage structure associated with it for storing information items relating to which design features have been applied to which document databases. The method comprises a procedure of controlling a development process of design elements or design features so that these are marked as user–reserved during the development process.

GENERATION AND MAINTENANCE OF DOCUMENT DATABASES

The present invention concerns a method of generating and maintaining software
applications comprising one or more document-oriented databases from a common library

5   comprising a number of standard design features and common architecture that are
applied to the applications.

In particular, the invention concerns a method wherein a record is kept relating to which
standard design elements have been applied to which databases, so that an upgrade of

10 single standardising elements may be performed in all relevant databases within the
applications in an easy and fail-safe way, both automatically as well as manually.

BACKGROUND OF THE INVENTION

15 It is well-known to construct software applications using a plurality of databases, each
comprising an amount of data and/or references to data stored in other databases, and a
number of instructions or design elements regarding handling of the data and the
database, presentation of data, communication with other databases and applications
etc., and/or references to such instructions/design elements stored elsewhere.

20

One type of databases is relational databases, in which the instructions typically are
stored in one or more central, common libraries. Applications using relational databases
are generally easy to maintain and update because changes in instructions/design
elements for the application or set of applications only need to be effectuated in the

25 common library (or libraries). One drawback of the use of relational databases is that the
use of an application involves data exchange between the application and the common
library, which burdens the data communication elements involved and is time consuming.
Another consequence is that the applications need to be in constant or very frequently
connection via data communication elements to the physical storage site of the common

30 library. Finally, an overview of the use of the individual instructions/design elements in the
various databases of the applications is not automatically provided so that the
consequences of an update of a design element may be difficult to foresee, especially
when dealing with complex applications or sets of applications referring to the same
common library or libraries.

35

Another type of databases for software applications is document-oriented databases in which the instructions/design elements are stored within the same data structure as the data. Document databases are fast in use and the applications may be operated from sites with only temporarily data communication contact to central data storage facilities. A

5   drawback is that an update of a common design element requires that all databases within the application or the set of applications that interacts or co-operates need to be inspected in order to ensure uniformity of the elements in the databases. Commonly in the development of applications, the design elements are modified or tailored for a specific use in a database and this version of the design element exists only in the specific

10   database it was modified for and new databases constructed from that database. Update of such applications is highly time-consuming and the risk of overlooking design elements in the update, resulting in malfunction of the application, is high.

DESCRIPTION OF THE INVENTION

15

The present invention relates to the design and update of software applications comprising one or more document-oriented databases, in the following named document databases or just databases, typically databases following the standards of Lotus Notes. With the present invention, the advantages of the relational databases are transferred to

20   document databases without loosing the advantages of those. The method may be regarded as the application of an object oriented programming language to document databases.

It is an object of the present invention to provide a method for generating applications

25   comprising document databases in which a significant number of the design elements used in the databases are created by means of applying standardised design features from a common library.

It is a further object of the present invention to provide a method and a system in which a

30   documented record is kept of the application of standardised design feature in the creation of databases.

It is a still further object of the present invention to provide a method and a system in which the application of standardised design features to the document databases

comprised within the set of applications that are to be maintained with the use of the same common library is recorded in a common data structure.

It is a yet still further object of the present invention to provide a method and a system in

5 which the update of a standardised design element is performed with use of the aforementioned record, with the exception of specific design elements that have been marked for non-modification.

It is a yet still further object of the present invention to provide a method and a system of

10 controlling a development process of design elements or design features that prevents that more than one developer or group of developers work simultaneously on the same design elements or design features without being aware of the situation and so that design elements or design features may be temporarily protected from overwriting during a developing process.

15

Other objects of the present invention will be apparent from the description given below.

Thus, the present invention relates to a method for generating or maintaining a plurality of document databases comprising design elements and comprised within one or several

20 applications, the method comprising the step of standardising at least some of the design elements of the document databases by communicating design features to and applying the design features to each of the plurality of document databases from a common library.

The standardised design features in the common library may in a simple and preferred

25 from be the standardised design elements, that are copied to the database in question when it is applied to said database. In another embodiment, the design features are instruction of how to create the design element, e.g. by the use of samples of code from a code library.

30 The databases of the software applications are stored on computer-readable media such as e.g. magnetic discs or tapes, optical discs, CD-ROMS, RAM circuits, etc., each media being in permanent or temporarily data-communication contact with a computer having a central processing unit and input and output units. The databases may be stored on a plurality of media being connected to, or in contact with, a plurality of computers being

interconnected permanently or temporarily via a data communication network, which may include a local network, a wide area network, a public network or any combination thereof.

The common library of standardised design features as well as the relevant software for
5 carrying out the invention is likewise stored on computer-readable media.

A number of examples of design elements are given in the example below.

The present invention relates in particular to such a method that further comprises a
10 version control system, being a procedure of controlling a development process of design elements or design features, the procedure comprising the steps of

selecting at least one design element or design feature,

marking the selected design element(s) or design feature(s) as user-reserved in a central data storage structure pertaining to said plurality of document databases,
15 producing (a) new design element(s) or design feature(s) to replace the selected one(s),

undoing said marking of the selected design element(s) or design feature(s), and

replacing the previously selected design element(s) or design feature(s) with the respective new design element(s) or design feature(s),
20 the replacement of a design element or a design feature being prohibited if said design element or design feature is marked as user-reserved in said central data storage structure.

The mark of the selected design element(s) or design feature(s) may further comprises
25 data relating to a proprietor of the mark(s) and it is preferred that the step of undoing the marking only may be performed by the proprietor of the mark(s).

Another possible feature is that the step of replacing selected design element(s) or design feature(s) with the respective new design element(s) or design feature(s) may be
30 performed by the proprietor of the mark(s) even if the design element(s) or design feature(s) is (are) marked as user-reserved.

To ensure that the database and thus the application still function after an unsuccessful development process the procedure of controlling a development process preferably in
35 addition comprises the steps of

associating a label with a selected design element, a design feature or a document database and storing said label in a data storage structure, the label comprising sufficient data to restore the design element, design feature or document database with features in accordance with the features of the design element, design feature or document database

5   at the time of association of the label,

restoring a design element, a design feature or a document database by use of data comprised within a label associated with said design element or design feature, and

replacing the design element, design feature or document database with the restored design element, design feature or document database.

10

It is in general preferred that a technical foundation library is defined within the common library and comprises at least one design feature and each of the plurality of document databases comprises for each design feature of the technical foundation library a design element to which said design feature has been applied. The technical foundation will

15   preferably comprise a plurality of design features which is when applied to a database forms a set of techniques and objects used in all document databases within all applications, e.g. for creating the same look-and-feel of the different applications and for defining standards for input and output to and from the document databases and communication between the document databases.

20

The common library may have at least one data storage structure associated with it for storing information items relating to which design features have been applied to which document databases and the step of generating a document database should in that case include communication of information items relating to the application of design features

25   to the document database to said at least one data storage structure and saving the information in said at least one data storage structure.

With such information items stored, the method of the invention may comprise at least one update procedure constituting a maintenance step performed on the document

30   databases, each update procedure comprising the steps of

including a new design feature in the common library for replacement of an existing design feature of the common library from which the new design feature differs,

retrieving information items from the at least one data storage structure relating to which document databases the existing design feature has been applied to, and

applying said new design feature to all of the document databases to which said existing design feature had been applied according to said retrieved information items, except where replacement of the existing design element in the individual document database is contradicted by other information. Furthermore, the method may comprise

5   storing, in the at least one data storage structure, information items relating to which document databases the new design feature has been applied to.

The method may additionally or alternatively comprise at least one update procedure constituting a maintenance step performed on the document databases, each update

10 procedure comprising the steps of
        including a new design feature in the common library,
        applying said new design feature to all or some of the document databases, and
        storing, in said at least one data storage structure, information items relating to
    which document databases the design feature has been applied to.

15

As a further alternative of addition, the method according to the invention may comprise at least one include procedure for including a previously generated document database into the plurality of document databases maintained from the common library, the previously generated document database comprising at least one design element to which at least

20 one of the design features of the common library has been applied, the include procedure comprising the step of storing information items about which design features have been applied to said document database and saving the information item(s) in said at least one data storage structure.

25 As an enhancement of the update procedure for updating design elements of the databases, protection information items relating to at least some of the applications of design features to the plurality of document database may be stored, each of the stored protection information items regarding whether updating of a specific design feature applied to a specific document database in a subsequent update procedure performed on

30 the databases is admissible, said protection information items being used by the at least one update procedure to determine whether replacement of existing design features in individual document databases is contradicted. The protection information items may be stored in the at least one data storage structure associated with the common library and/or the protection information items may be stored within the individual document

35 databases, depending on the organisation of the applications and the common library.

To make the individual document database at the same time updateable and individually configurable, the operation of a plurality of said design elements being standardised by applying design features from the common library comprises in an advantageous

5 embodiment of the present invention the steps of

activating an exit of the design element,

retrieving a possible feature being associated with the specific exit from at least at least one setup definition defining features of the individual document database to be applied to design elements of the individual document database, and

10 applying said possible feature.

In particular, at least one of said plurality of design elements retrieving features from the setup definition may be an event monitor being associated with and applied by at least one of the design elements of the document database to which design features of the

15 common library has been applied, the operation of the event monitor comprising the step of monitoring each possible event associated with the design element by which it is applied, and in response to the occurrence of one or more of said possible events the steps of applying one or more possible features associated with said possible event and further in response to one or more of said possible events activating an exit associated

20 with said possible event. The event monitor comprises in a most preferred embodiment exits associated with each of said possible events.

In a particular and preferred embodiment of the present invention, a set of document types are defined and each document within the document databases contains

25 information about its specific document type from a set of document types, the document type being significant for applicability with respect to the access and use of the document, and each design element of the plurality of document databases for accessing and using documents also contains information about the specific document type which may be applied to it, the document type definition comprises the definition of a plurality of data

30 fields with respect to the name and contents of each of the data fields, each document of a given document type comprises said plurality of data fields of the definition.

The document types are defined locally in each document database, but at least one and preferably a plurality of document types of the set of document types may be defined

35 commonly for the plurality of databases.

For the above-described method, the document databases are preferably document databases generated according to the standards of Lotus Notes or a modification or further development thereof.

5

The mentioned data storage data structure(s) is (are) also stored on computer-readable medium or media and the various procedures are effectuated by computer software designed for that purpose, the software being stored on a computer-readable medium.

10   The above-mention procedures and features may be combined in any reasonable manner according to the invention and do not necessarily depend on each other.

The present invention also relates to a method for performing an updating procedure on a plurality of document databases comprising design elements and comprised within one or
15   several applications, of which design elements at least some comply with design features from a common library, the method comprising the steps of
        including a new design feature in the common library for replacement of an existing design feature of the common library from which the new design feature differs,
        retrieving from at least one data storage structure associated with the common
20   library information items about which document databases that comprise a design element which complies with the existing design, and
        applying said new design feature to all of the databases that comprises a design element which complies with the existing design feature according to said retrieved information items, except where replacing of the existing design element in the individual
25   database is contradicted by other information.

The above discussed features and procedures of the first-mentioned method of the invention may be applied to this method according to the invention. However, this method differs for the first-mentioned in that it also relates to document databases that already
30   have been formed, e.g. by a software company, and is maintained and/or updated on a computer system being external to the one where it was generated.

Furthermore, the present invention relates to a system comprising a plurality of document databases comprised within one or more applications and residing on data storage means
35   of a computer system comprising one or more computers having data communication

enabled there between permanent or temporarily, each of the document databases comprising at least some design elements which comply with design features from a common library, the system comprising at least one data storage structure containing, for each of the design features of the common library, information about which of the plurality

5 of document databases that comprise a design element which complies with the design feature.

Again, the above discussed features and procedures of the first-mentioned method of the invention may be incorporated in the system according to the invention

10

Finally, the present invention relates to the version control system discussed above which may be regarded as an invention in itself. Thus, the invention further relates to a method of controlling a development process of a plurality of document databases comprising design elements and comprised within one or several applications, the method comprising

15 the steps of

selecting at least one design element,

marking the selected design element(s) as user-reserved in a central data storage structure pertaining to said plurality of document databases,

producing (a) new design element(s) to replace the selected one(s),

20 undoing said marking of the selected design element(s), and

replacing the previously selected design element(s) with the respective new design element(s),

the replacement of a design element of the plurality of document databases being prohibited if said design element is marked as user-reserved in said central data storage

25 structure.

EXAMPLE

In the following an example is given of how the present invention can be used in a specific

30 software toolkit for developing applications and parts are given with direction for users to illustrate the functionality's of such a toolkit according to the invention. The example is illustrative for a number of features of the invention and is not in any way limiting the scope of the invention.

Lotus Notes is a document-oriented database system, which is exceptionally good at keeping track of all forms of electronically available objects. Through its built-in Workflow capabilities, it is also very good at keeping track of documents through their entire lifetime. Using a Domino server, you can extend these capabilities to work together with the

5   Internet World Wide Web (WWW). The following are examples of good applications for Notes/Domino:

- Electronic mail and calendering (standard application).
- Library applications, edited infrequently but full-text indexed for searching, and
10   categorised for rapid access. High security can be introduced with certain documents only readable by certain classes of users.
- News applications (added/updated daily) - frequently imported from external sources.
- Workflow applications where a multi-sectioned document is edited by defined users in turn, to create, approve and implement work or resource requests.
15 • Action tracking applications, where managerial supervision is important.
- Discussion databases, where communication of a 'many-to-many' is processed and many more.

The Software Development Kit (SDK) is used to build, maintain, expand and upgrade
20   applications composed from document databases within the standards of Lotus Notes. By providing the users with this toolkit they are able to install and modify the standard applications, and are further enabled to build their own applications on top of the standard application architecture, thereby reducing the development time and cost.

25   In short, the SDK provides the following:

- An automated installation and configuration process
- An extra abstraction layer upon Notes/Domino
- One set of objects which can be used in several applications
30 • Easy development of Notes applications - including standard objects and methods

The SDK comprises the following functionalities:

- Centralised Database Administration Panels
35   • Configuration Wizards on several levels:

1. For all databases
2. For selected databases
3. Per database
4. For selected setup documents within a database (Super user)
5. Rule-based Workflow
6. Event logging
7. Document archiving
8. Action Tracking

Standard objects include:

- Mail/Calendar integration
- Telephone API integration
- On-line ODBC Access to ERP systems
- Correspondence including Notes letters and OLE Objects
- SMS (Short Message Service) integration
- Form Styles (background colours and graphics)
- View Styles (colours and fonts for different categories and documents)
- View Action buttons and more
- Access Control Lists (ACL)

The standard objects have the following features:

- A consistent and innovative look & feel
- Can be tailored to a specific line of business (OEM) or customer needs
- Multilingual support (glossaries currently exist in English, Norwegian, Swedish, Danish and Finnish)
- Internet and Intranet enabled
- Can be upgraded (even though customer specific changes have been applied)
- The Application Architecture provides a set of rules/techniques for all aspects of Notes/Domino development
- Documentation including Design & Technical standards
- Guidelines for installing, deploying, building and tailoring applications

The SDK consists of the following elements:

- Installation/Upgrade Wizard
- Design Container

- Design Repository
- Design Analysis
- Guidelines

5 Furthermore, the following add-on tools may be provided:

- Design Analysis
- Translation
- Version Control
- Migrator

10

And lastly the Application Architecture, which again consists of the following elements:

- Technical Foundation
- Common Databases
- Repositories
15 - Standard Objects
- Design & Technical Standards

Each of the above elements is explained in the following sections.

Installation/Upgrade Wizard

20 The Installation/Upgrade Wizard makes it easy to install or upgrade all or part of the framework: "install and go" as well as create a customer-specific installation.

The Wizard features 2 modes:

- The configuration mode, used when generating a specific CD to an End-User
25 - The installation mode used to install the applications located on the CD

The Configuration mode of the Wizard features the following:

- Possibility to hide the design
- Packs the databases into one file
30 - On a whole, makes a version ready to be burned onto a CD

The actual Install mode of the Wizard features the following:

- Generates new copies of the databases

WO 00/22547   PCT/DK99/00539

13

- Ensures that all database references from one database to another are modified according to the server location and replica-ID of the installed databases
- Enables Name & Address Book integration to the Staff database of the Framework
- Creates groups referenced in the ACL in the Name & Address Book
- Enables references to the Agent Log database, which is used by all scheduled background agents
- Automatically fills in the Database Catalogue
- Sign all agents, so they can execute on the server where they are being installed
- Makes sure that all standard Notes Setup documents are saved on the corresponding Profile documents (used for performance reasons)
- Gives the user the option of starting up the Configuration Wizard, which then enables the user to perform the post-installation requirements using an intuitive UI
- If the system is being upgraded from a previous version, all elements (design elements and setup documents) will be upgraded on a per element basis. If any changes have been made to the standard version, these elements will not be updated, new databases are added where necessary, old databases are not removed

Design Container

The Design Container is the point of control for the objects delivered together with the SDK. The Design Container features the following:

Building Blocks

When a database has been created, the building blocks can be used to assist the building/programming of that database, and thereby ensure that all developers are following the same technical concept and architecture. Building blocks, which can be used when designing a database, includes the following:

- Code elements, which include standard coding - for example how to make a standard Keyword Lookup. The SDK is delivered with approximately 20 standard Code Elements, which can be expanded by the User.
- Form Body, which can be any part of a Form - for example a standard address block that should be used in all applications where an address has to be specified. The SDK is delivered with approximately 20 standard Form Bodies, which again can be expanded by the User.

- Graphics, which could be used in a Form or a Navigator - for example a graphic indicating that a field is required for input. The SDK is delivered with approximately 10 standard pieces of graphics, which can again be expanded by the User.

5 Objects

The Design Container controls a set of objects, which are used in one or more applications built on top of the Application Architecture, and can be used when creating new databases. The objects controlled by the Design Container include the following:

10 • Design Elements including objects for different standard functions at all levels - for example workflow, mail and calendar functions, graphical functions, and toolbars. Objects of the Design Element type can include any combination of Notes Design Elements and Setup documents - for example the Direct Mail objects which include 1 Form, 3 Views, 6 Agents, 2 Script Libraries and 4 Keyword Setup

15 documents. The design elements referenced can either be placed in the Design Repository or in another Notes database (Object database). The SDK is delivered with 15-20 Object databases used only as placeholders for the object definitions kept within the Design Container.

- View Action buttons including Action buttons to be used in many or all applications

20 - for example the Create button used by the Application Architecture, which is the same in all databases, and located within all Views - but only maintained in one central place.

- Form Styles, which can be used to modify the background colour and/or graphics on all or selected Forms within one or more databases.

25 • View Styles, which can be used to modify the background colour and/or fonts and sizes on all or selected Columns in one or more databases. For example changing the way the first Category in a View appears in all databases.

- Access Control Lists (ACL) for controlling users access to databases, both with respect to viewing of documents, modification of documents and modification of

30 the design elements of the database.

Databases

The highest level controlled within the Design Container is the Database level. It is used to define which objects (Design Elements, View Action buttons, Forms, ACL's and View

35 Styles) should be used in which databases, so that whenever an object (or an object

definition) is changed, the databases using that object will be changed automatically, either by using a scheduled background Agent located on the Server or by manually using the "Push" technology.

5   "Pushing"
The key feature of the Design Container is the method of "pushing" the design into one or more databases. Doing this means the objects can be maintained in the Design Container or any of the Object databases, and then "pushed" to several databases using these objects.

10

The same applies to Setup documents (Keyword lookups, database lookups etc.), which can also be maintained centrally in the Design Container or the Object databases and then "pushed" to many databases at the same time.

15 The Design Container is also used to generate a new database based upon standard design objects, thereby linking that database to the objects included. Through a Wizard, the user is guided through the creation process and, based upon the selected objects, a database entry will be created in the Design Container, and thereby enabling future changes to one of the objects in question to be reflected into the new database.

20

Lastly, the Design Container also features a "Pick element" option, which can be invoked when designing a database. Using this feature, a database can be built using standard objects or elements from the Design Container, which can be any Notes object (Form, View, Agent, Script Library etc.), specific piece of programming language (Script or

25 formula), part of a Form or graphics (building blocks).

The design container with the design repositories comprising standard design elements and the push/pick functions is illustrated in Fig.1.

30 Design Repository
The Design Repository is used as a placeholder for all Form Body definitions and some Design Element objects located in the Design Container. The Design Element Objects kept within the Design Repository typically only include one or two design elements.
If an object definition in the Design Container consists of many design elements and setup

35 documents, these elements are typically kept in a separate Object database (repository).

Design Analysis

This database is used to collect information from all or selected databases available on the server. The Design Analysis includes information about ALL design elements located

5   in a database. Another purpose of the Design Analysis is to extract all pieces of coding located in a database, so developers can easily look for a specific piece of programming by using FT-search, or places where, for example, a variable or a Script Library routine is being referenced.

10 Many of the properties for a given design element are also included in the extract of information, so the Design Analysis can also be used for QA purposes. For example, to check that the 'Do not allow Design Refresh/Replace to modify' flag is not set on any element.

15 Another possibility for QA on the applications is to generate a QA-record where the user can specify certain rules the Analyzer should run against the data extracted.
For example, the user could generate a QA-record, which specifies that the Analyzer should search all Forms, and check that all naming conventions have been followed. The QA-records are completely rule-based, and can therefore be configured by the individual

20 user.

The Design Analysis database can also be used for maintenance purposes as old elements or old versions of elements can be removed from the source databases, even though the user is located in the Design Analysis database. The Application Architecture

25 includes a version number used on each Design Element, and this version number is also reported by the Design Analysis database.

Translation

The Translation database simplifies the task of translating Notes databases. The

30 Translation database relies on Notes Global Designer to translate the entire database, except scripts and documents. Translation of scripts and documents are handled by the Translation database. Translating one or more databases can be carried out by going through the following steps:

- Defining reference language, path to Notes Global Designer and Setup document translation options. This is done in the Global Options document. Setup document translation options tell the build-in document translator how to treat Setup documents.

5
- Creating a Translation document. The Translation document defines the glossary to use, the target language and all Notes Global Designer settings.

- Creating a Database document for each database to be translated. Database documents can be auto-created by using settings from the Global Options and the Translation document. Database documents define source and target database,

10
and what type of documents to translate.

- Running the glossary builder. The glossary builder adds text from scripts and documents to the glossary. It finally launches Notes Global Designer to do the remaining translation. Notes Global Designer runs without user interaction.

- Translating the glossary. The Translation database comes with several built-in

15
tools to make this task quick and simple.

- Running the database builder. Builds the target database(s) from the glossary.

Features
- Notes Global Designer compatible glossary and setup in a single database

20
- Easy setup of all Notes Global Designer options. This is done once for several databases

- Advanced document translation setup options for each database

- Global Setup document translation options

- Use internal or external glossary

25
- The possibility to translate only documents

The Translation database has been used to translate the "Business Application Framework" into 5 new languages, and there are more to come. The Framework currently consists of 70 databases with more than 12,000 design elements and several thousands

30 Setup documents.

Application Architecture

Technical Foundation

The Technical Foundation is a set of techniques and objects used in all databases within

5 all applications. It includes methods for developing Notes/Domino applications. This

foundation assures that all applications are built over the same template, and that they will

all have the same "look and feel". The Technical Foundation possesses the preparation

for translation via the Translation database (described earlier).

10 The basis of the Technical Foundation is provision of generalised Administration panels,

where most things within a database can be maintained from:

- Keyword and database lookups
- What fields to inherit to/from in which forms

15 - Which category levels to show in Views
- Which Forms to show in Views
- Scheduling of Agents
- Maintenance of logically deleted documents
- Computed sub-forms to define the header, footer, toolbar, Reader & Author names

20 and more...
- Which Navigators and Views to use
- How to log/archive
- How to build up a homepage
- How to build up the individual WEB pages

25 - What graphics/buttons to show on the WEB and where they should be located
- And much more...

The Technical Foundation is the lowest level of any database, and should always be

included as an object, whenever a new database is generated by using the SDK.

30

The Application Architecture is built for the purpose of the Customer tailoring the

applications according to a specific line of business or customer requirements.

Common Databases

All databases/modules built using the SDK on top of the Application Architecture can use the Common Database, which forms a part of the Architecture.

5   The Application Architecture includes the following Common Databases:
- The Staff database
- The Customer database
- The Products database
- Log
10   - On-line Help
- Standard Letters & Phrases
- Common Calendar
- Common Contacts
- Help Template
15   - Homepage

The Staff database is where all system users must be defined. It automatically exchanges information with the Notes NAB.

20   The Customer database is where all the customers referenced in the system(s) must be defined.

The Products database is where all the products referenced in the system(s) must be defined.

25

The Log database is a centrally located database, which is used to store information based on the usage of the individual databases. For each database built on top of the Application Architecture, the Administrator can specify which events should be logged (if any).

30

The following events can be logged:
- Database Open
- Database Close
- View Open
35   - Document Open

- Document Close
- Document Save
- Document Delete

5 Not only does logging work when the user is located at the office, but also when the user is off-line, as logging is done locally in the database and, once replicated to the server, a background Agent will automatically transfer the Log information to the centrally located Log database.

10 The On-line Help database is a centrally located database used to provide end-users help with individual applications. Help can either be invoked using a special entry in the Navigators directly from a form using a special Action button, or the user can enter the Help database directly. The Help database is created in such a way that the Customer can enter additional help for users.

15

Help is provided on the following levels:
- Application Overview
- Database Overview
- Navigators within a database
20 - Views within a database
- Forms within a database
- Quick Guides for the most common tasks within each database

Standard Letters & Phrases is a centrally located database containing all Letter-templates
25 and phrases to make up the letters. It can be used from within all applications built on top of the Application Architecture. The letter templates and phrases are based upon tagging, so the individual Notes or Phrase letters can be merged with the data of the documents where the correspondence is written from (for example a Contact Profile). The database also supports the use of OLE-objects, so if the Customer wants to use MS Office or Lotus
30 SmartSuite instead of Notes correspondence, that can be handled as well.

The Common Calendar is not really like the rest of the Common Databases, which are typically referenced from the individual databases built on top of the Application Architecture. However, it is an essential and crucial part of any end-user environment, and
35 is therefore always delivered when any of the standard applications is bought.

The Common Calendar includes the following features:

- Standard Common Calendar functionality
- 'Up to the minute' update of calendar appointments from the users individual mail/calendar files
- Views for all or selected employees
- Complete synchronisation between the individual employee's calendar and the Common Calendar
- Calendar views on 'own' or NAB groups
- Booking of freetime directly from the Common Calendar
- Automatic deletion of old appointments

The Common Contacts database is a common entry point to other databases used to manage contact information such as customer contacts and supplier contacts.

The Help Template database is used for creation of customer-specific Help databases.'

The Homepage database is the start-up pages of the Business Suite, the Software Development Kit, the Tools and the Management Information System. It is thus always available on all system, even though some of the hot-spots may not have relevance if the module or product concerned is not part of the installation.

A number of additional databases may also be available:

- Discussion
- Downloads
- Handbook
- News

The Discussion database allows employees to discuss issues, for example marketing matters or company improvement suggestions.

The Downloads database allows for the possibility to place computer files for download on a company's homepage on the Intranet and/or Internet.

The Handbook database manages sets of documents categorised in sections and chapters. For example, it may be used to store information or guidelines for employees in a company or product guide. to ensure that only appropriate information is stored in this database, each document must be approved by an authorised person before being
5   available for public access.

The News database provides employees with the latest news within a business, for example from the marketing department. It can be used to communicate marketing strategies, new ideas and even publish product and press releases.

10 **Standard Objects**

The Standard Objects included as part of the SDK, include the following:
- Workflow
- Action tracking
15
- Mail merge
- Direct Mail
- E mail

All of these objects can be used when generating new databases, or when adding
20 functionality to an already existing database.

The Workflow Object is a general feature, which can be added to any form in any database built on top of the Application Architecture without changing the design of any of those forms. Using a general setup feature, any form can be workflow-enabled and
25 submitted to a serial or parallel review and/or approval process using Notes Mail. During this review process, access to the document is controlled according to the document's current position in the review cycle. The Workflow Object includes many other options - for example starting new workflow upon completion of a workflow, controlling what fields can be modified during an approval/review process and much more.

30

Like the Workflow Object, the Action Tracking Object is also a general feature that can be added to any database built on top of the Application Architecture. Implementing the Action Tracking Object means that Actions (or Tasks) can be generated based on the contents of a specific form or by choosing a manual creation of an Action. The Action is

integrated with the Task form in the Notes Mail file. Once marked as completed in the mail file, the corresponding action in the source database will automatically be marked as Complete as well, or vice versa.

5   Mail Merge is to be used with the Common Database "Standard Letters & Phrases" and can be used to write letters from all applications built on top of the Application Architecture. The letter templates and phrases are based upon tagging, so the individual Notes or Phrase letter can be merged with the data of the document where the correspondence is written from (for example a Contact Profile).

10

The Direct Mail object is an extension to the Mail Merge object and can be used to generate direct mails to number of contacts - all at once. It can be implemented into any database built on top of the Application Architecture.

15 The Design & Technical Standards include visual and programming standards, which should be used when creating a new database. The Design Standards can be described as a handbook to follow, with all necessary information on how to change font, layout and other features on a given entity of a Notes database (Form, View, Navigator etc.). The Technical Standards are programming guidelines, when an existing based database is

20 extended or a new one created.

The Design & Technical Standards include information in the following areas:
- General Overview
- Technical Overview

25
- How does a database work
- What events and programming techniques do we use
- Design Overview
- Designing databases
- Designing Forms

30
- Designing Views
- Designing Agents
- Designing Navigators

Forms

All forms are built on subforms. Any part of a form, which is not specific for the application, must consist of one or more subforms. Text fields specific for the form are placed directly

5   on the form because the amount of time it takes to open a form is affected by the number of subforms. One vital exception is the hidden fields placed right at the top of every form before any subforms are embedded.

Guidelines

10   The following Guidelines apply to all subforms:

- A subform shall always start with a line of text "Start of <SubformName>". This line shall be hidden for reading, editing and printing, but is visible in the form where it is embedded. Otherwise it is impossible to read what subforms are placed where.
- A Subform shall hold fields containing data, which is associated together in some way.

15   • The contents of a Subform shall follow the same guidelines as those for a form itself.

- Actions appear in the action bar in the order the subforms appear in the form - but these are
- restricted to only two in number:
  - .SysStdToolbar.

20   ▪ .AppToolbar.

- A subform shall finish with the label "End of <SubformName>". This line is also hidden under all conditions.

Standard subforms

25   The following chapters describe some of the standard Subforms in the Design Container. Beyond these subforms, there will naturally be built further standard subforms as development goes forward. For a subform to be considered as a Standard Subform in the Design Container, the following rules will apply:

-The planned Subform shall be general in its use, and must not contain any fields which

30   are application specific. The Subform shall always follow and be updated to the current valid standards as laid down in this book.

- The Subform shall be planned to be used in several other applications.

After the Subform has been placed in the Design Container, it shall be documented

35   herein. This shall consist of:

- Short description of the purpose of the subform.

- Description of which databases the subform is used in. Also, it shall be recorded every time it is used in a new database.

- Short description of the fields, event scripts and actions included in the subform.

5

Creating a new database

The "Create Database" button triggers an action which invokes an agent ".ElementCreateNewDatabase". This in turn invokes four agents in succession:

10

.ElementCreateDatabase

.ElementSelectElements

.ElementCopyDesignElement

.ElementUpdateKeywords

15

These will now be examined in greater detail:

ElementCreateNewDatabase

This agent has the task of actually creating the new database, although on completion it is

20 still empty. It also creates an Environment variable containing the server name and file name of the new database. This is passed as a 'parameter' to the other agents in the above list. The agent runs the following steps:

- Display a dialog box to receive the new database server name, title and file name. These must be entered following the Mandatory standards for naming and version

25 number.

- Check if the database already exists, and if so does the designer want to delete it. If not, the agent exits.

- Create the new database on the designated server and add it to the designer's desktop.

30 - Set the database title and the environment variable "EnvDstDb" = .Server , .FileName.

.ElementSelectElements

This agent presents a dialog box where the designer can select which standard elements he/she wishes to be included in the new database. Currently this list is:

- Complete Technical Foundation.
5 - Update Technical Foundation.
- Action tracking.
- Banners: Administration, Computer, E-Mail, Group, Heading, Help, Keyword, Mail, Pie
- Chart.
- Complete Mail Merge.
10 - Update Mail Merge.
- Main Form.
- Response Form.
- Update info (Server).
- Database script.
15 - Merge.

The selected list is preserved in the designer's ".Parameters" profile document under field name "Elements".

20 **.ElementCopyDesignElement**

This agent retrieves the list of selected elements from the designer's ".Parameters" profile document. For each element in the list, it reads its Design Element document. This document holds the source server name and file name where the components reside, and a number of fields listing the components, which make up that particular element. Each

25 component is listed by name and its DocumentUniqueID. For each Design Element document, the agent calls subroutine "CopyToDatabase" which resides in the script library ".CopyDesign". This subroutine opens the source database, and for every component listed, finds its 'DocumentUniqueID' and copies it from the source to the destination database. These actions are achieved partly with subroutines in the same library

30 ("DoMove", "Scan", etc.) and partly with C-language subroutines contained in the attached Dynamic Link Library (DLL) "rio.dll". The entry points to the C-language subroutines are defined in the Declarations section of the script library.

.ElementUpdateKeywords

This final agent actually initiates running another agent ".KeywordsUpdate" in the
destination database. ".KeywordsUpdate" and the script library ".Keywords" have been
placed there by the previous step (.ElementCopyDesign). ".KeywordsUpdate" in turn calls

5 three subroutines in the script library ".Keywords":

- Call UpdateKeywords.
- Call CleanupKeywords.
- Call CleanupDatabases.

10 **"UpdateKeywords"** has exactly the same effect as activating the action button "Update
Keywords" in the destination database administration view. It copies the whole set of
standard keywords lists from all setup documents using form ".SetupKeyword" into a new
profile document ".Keywords". It also copies into another new profile document
".Databases" the database synonyms found in setup documents based on form

15 ".SetupDatabase". These are typically for the four key supporting databases: Design
Container, Help database, activity Log database and Agent log database.

CleanupKeywords and **CleanupDatabases** are safety routines which check **all**
keywords and database synonyms in the two profiles, and remove those which are **not**

20 now in the. SetupKeyword or .SetupDatabase documents. They might have been copied
in with a previous design selection, and are now no longer needed. On completion of
these four steps, the destination database contains all the source code needed to run the
selected Design Elements, and two vital profile documents have been populated. The
designer can now open the new database, and get on with the main task of designing the

25 forms and views for the user application.

Version Control System

The Version Control System enforces a disciplined and thus systematic approach in
database development activities. This is performed by means of check-in and check-out

30 procedures for databases, their design elements, user documents and Setup Documents.
Checking-in a database, or element, implies that it is available to other system developers
for further development. Checking-out of a database, or element, implies that it is 'locked'
for the specific developer concerned. To achieve this the Version Control System provides
of two types of databases with various version control documents: Version Control

35 Configuration and Version Control Log.

The Version Control Configuration is a database that manages the status of each database that has been checked-in to the version control system and this via a *Database Configuration Document*. This document is categorised in accordance with a user-

5   specified project, namely that defining the purpose for including the database concerned into the version control system. It includes a reference to the Version Control Log database containing the *Element Documents* for the database concerned. Only one Version Control Configuration database exists in an installation.

10 The Version Control Log is a database that manages the status of each design element, user document and Setup Document for those databases that have been included in the Version Control Configuration database. This is performed via an Element Document for each of the elements concerned and is updated for each version thereof. The Element Document is updated with the changes made to the element concerned. It is possible to

15 have multiple Version Control Log databases in a single installation.

The relationship between documents and databases is seen in Figs. 2 and 3. Here, the single Version Control Configuration database contains Database Configuration Documents which refer to the respective Version Control Log database concerned. Each

20 Version Control Log database has one or more Database Definition Documents, namely for managing the databases whose elements it controls. There is associated one or more Element Documents with each Database Definition Document.

It is also possible to associate two or more elements from the same database in an Object

25 Definition. These allow management of collections of elements, namely collective check-in and collective check-out. It is furthermore possible to associate two or more Object Definitions to from a Group. These allow management of groups of Object Definitions, namely collective check-in and collective check-out.

30 Groups are not restricted to the same database. The Object Definitions they include could be from different databases.

The following sections provide a detailed description on using the Version Control System and its associated functionality.

35

Using the version control system

A number of decisions are to be taken when implementing the version control system.

Step I - Decide whether the database of origin is to be used

5   This determines whether the element's database of origin is used as the 'owner' of the element concerned. If the database of origin is to be applied, then this affects the order in which databases are to be checked-in to the version control system. This requires following all the steps outlined below. If the database of origin is to not used then only Steps IV and V are of relevance.

10

Step II - Check-in the database of origin

- Identify the database of origin
- Check-in the database of origin to the version control system

Databases of origin are checked-in according to the number of elements they own in

15   Destination Databases. The source of the largest number of elements is first checked-in.

- As this is the first check-in to the version control system, select Use origin in the Version Control System Setup dialog.
- Once checked-in, remove those elements from the version control system that are not to be owned by the database of origin. Elements which are removed are still known to

20   the version control system, but await check-in of another database which includes these elements and will thus own them.

This is used where two origin databases have the some identical elements and it is not necessarily the first that is checked-in which is to be the owner of the element(s).

- Using the Design Container's push functionality, push the elements from the database

25   of origin to the respective Destination Databases. The elements in the Destination Databases are now stamped as originating from the database of origin, for example Technical Foundation in the architecture.

Step III - Check-in other databases of origin

30 - Repeat the approach in Step II for other origin databases. For example, in the architecture this would imply checking-in repositories such as the Design Repository.

- Identify the Object Definitions in the Design Container which use the elements from these databases of origin.

- Push the Object Definitions concerned to the Destination Databases. These elements

35   are stamped as originating from the database of origin, in this case the repository.

The elements in the repository that have already been checked-in via another database of origin, for example the Technical Foundation, have already been stamped and thus marked as owned by the database of origin, namely the Technical Foundation. If these elements have previously been removed they will then be owned by the repository as it is
5   checked-in.

Step IV - Check-in all remaining databases
Check-in the remaining databases. All database elements that do not belong to a database of origin, if this approach is used, will be stamped as belonging to the
10   Destination Database concerned.

Step V - Check-in and check-out of elements
Check-in of a database, whether it is a database of origin or not, implies that the elements which it owns may be checked-in and checked-out using the standard Version Control
15   System dialog. It is also possible to perform a series of functions on the elements. Roll-back functionality allows restoring previous versions as the current version and a history function provides a complete overview of the versions of the element concerned. Finally, if all elements in a database have been checked-in, it is then possible to label the database and the elements therein. This label allows reestablishment of the database in
20   accordance with the versions of its elements at the time of labelling.

It is important to distinguish the difference between a project and a label. A project, as mentioned earlier, is a broad classification including one or more labels. An example is where development of a database on a new user interface requires a new project, while
25   development or enhancement within a specific user interface only requires definition of new labels to manage database updates. If a database is to be applied to a new project after having been assigned to an earlier project it is necessary to copy the database and apply this new copy to the version control system as a completely new database. Labelling a database, on the other hand, is a continuum within a specific project.
30
Setup definition

Each document database that is generated by means of the present SDK and any document database that is in accordance with the design architecture of the present
35   system comprises a setup definition in which modifications of standard design elements

applied to the document database from the Design Container are specified. These modifications are specific for the individual document database and are used to adapt the standard design elements to the particular requirements for the function of the individual database. The setup definition of the database is not changed when new definitions of the
5   standard design elements are pushed from the Design Container and the modifications are thus preserved. This makes the document database upgradable without disturbing or disrupting the function of the document database.

The standard elements comprises a number of *exits* at which the standard design element
10 performs a look-up in the setup definition of the document database to see if a modification is specified and, if such modification exists, applies the modification.

The modification may be replacing a particular feature, such as a toolbar or a header, with another toolbar or header, or the modification may be an addition, such as an extra action
15 button to a standard toolbar.

Another type of modification is UserViewEvents that are stored in the Script Library of the setup document. When a view which is a standard design element is opened, an event monitor is activated and monitors each possible event connected to the view. These
20 events may be the opening or closing of the view, click or double-click of an item or region, drag-and-drop, cut, paste, recalc of calculation fields etc. For each event, the event monitor activates an exit and activates a script of the standard design element for the particular event, AppViewEvent as well as a UserViewEvent of the Script Library for the particular event. The UserViewEvent may be empty or may contain a modification of
25 the standard design element. The modification may be the appearance of e.g. an option menu, a warning, a help text or a dialog box or the modification may be a prohibition or a modification of an action the user is about to perform.

Document type

30

Each document of the document databases has a data field in which the Document Type, DocType, of the document is defined. The Document Type is a higher abstraction or classification of forms and defines the name and contents of a number of data fields of the document. The document may comprise a number of additional data fields, but all
35 documents of a particular Document Type comprises the data fields defined by the

Document Type. This means that the document can be interpreted by multiple design elements such as different forms and view lookups, as long as these are of the same Document Type.

5   The creation of a new form or other design element that is accessing data fields of the documents is thus enabled without extensive manual coding of the view(s) concerned.

CLAIMS

1. A method for generating or maintaining a plurality of document databases comprising design elements and comprised within one or several applications, the method comprising
5 the step of standardising at least some of the design elements of the document databases by communicating design features to and applying the design features to each of the plurality of document databases from a common library.

2. A method according to claim 1 comprising a procedure of controlling a development
10 process of design elements or design features, the procedure comprising the steps of
selecting at least one design element or design feature,
marking the selected design element(s) or design feature(s) as user-reserved in a central data storage structure pertaining to said plurality of document databases,
producing (a) new design element(s) or design feature(s) to replace the selected
15 one(s),
undoing said marking of the selected design element(s) or design feature(s), and
replacing the previously selected design element(s) or design feature(s) with the respective new design element(s) or design feature(s),
the replacement of a design element or a design feature being prohibited if said design
20 element or design feature is marked as user-reserved in said central data storage structure.

3. A method according to claim 2, wherein the mark of the selected design element(s) or design feature(s) comprises data relating to a proprietor of the mark(s).
25

4. A method according to claim 3, wherein the step of undoing the marking only may be performed by the proprietor of the mark(s).

5. A method according to claim 3 or 4, wherein the step of replacing selected design
30 element(s) or design feature(s) with the respective new design element(s) or design feature(s) may be performed by the proprietor of the mark(s) even if the design element(s) or design feature(s) is (are) marked as user-reserved.

6. A method according to any of claims 2-5, further comprising the steps of

associating a label with a selected design element, a design feature or a document database and storing said label in a data storage structure, the label comprising sufficient data to restore the design element, design feature or document database with features in accordance with the features of the design element, design feature or document database
5 at the time of association of the label,

restoring a design element, a design feature or a document database by use of data comprised within a label associated with said design element or design feature, and

replacing the design element, design feature or document database with the restored design element, design feature or document database.
10

7. A method according to any of the preceding claims, wherein a technical foundation library within the common library comprises at least one design feature and each of the plurality of document databases comprises for each design feature of the technical foundation library a design element to which said design feature has been applied.
15

8. A method according to any of the preceding claims, wherein the common library has at least one data storage structure associated with it for storing information items relating to which design features have been applied to which document databases and the step of generating a document database includes communication of information items relating to
20 the application of design features to the document database to said at least one data storage structure and saving the information in said at least one data storage structure.

9. A method according to claim 8 comprising at least one update procedure constituting a maintenance step performed on the document databases, each update procedure
25 comprising the steps of

including a new design feature in the common library for replacement of an existing design feature of the common library from which the new design feature differs,

retrieving information items from the at least one data storage structure relating to which document databases the existing design feature has been applied to, and
30   applying said new design feature to all of the document databases to which said existing design feature had been applied according to said retrieved information items, except where replacement of the existing design element in the individual document database is contradicted by other information.

10. A method according to claim 9, further comprising storing, in the at least one data storage structure, information items relating to which document databases the new design feature has been applied to.

5   11. A method according to any of claims 8-10 comprising at least one update procedure constituting a maintenance step performed on the document databases, each update procedure comprising the steps of

including a new design feature in the common library,

applying said new design feature to all or some of the document databases, and

10   storing, in said at least one data storage structure, information items relating to which document databases the design feature has been applied to.

12. A method according to any of claims 8-11, comprising at least one include procedure for including a previously generated document database into the plurality of document

15 databases maintained from the common library, the previously generated document database comprising at least one design element to which at least one of the design features of the common library has been applied, the include procedure comprising the step of storing information items about which design features have been applied to said document database and saving the information item(s) in said at least one data storage

20 structure.

13. A method according to any of claims 9-12, wherein protection information items relating to at least some of the applications of design features to the plurality of document database is stored, each of the stored protection information items regarding whether

25 updating of a specific design feature applied to a specific document database in a subsequent update procedure performed on the databases is admissible, said protection information items being used by the at least one update procedure to determine whether replacement of existing design features in individual document databases is contradicted.

30 14. A method according to claim 13, wherein the protection information items are stored in the at least one data storage structure associated with the common library.

15. A method according to claim 13, wherein the protection information items are stored within the individual document databases.

35

16. A method according to any of the preceding claims, wherein the document databases are generated according to the standards of Lotus Notes or a modification or further development thereof.

5 17. A method according to any of the preceding claims, wherein the operation of a plurality of said design elements being standardised by applying design features from the common library comprises the steps of

activating an exit of the design element,

retrieving a possible feature being associated with the specific exit from at least at

10 least one setup definition defining features of the individual document database to be applied to design elements of the individual document database, and

applying said possible feature.

18. A method according to claim 17, wherein at least one of said plurality of design

15 elements is an event monitor being associated with and applied by at least one of the design elements of the document database to which design features of the common library has been applied, the operation of the event monitor comprising the step of monitoring each possible event associated with the design element by which it is applied, and in response to the occurrence of one or more of said possible events the steps of

20 applying one or more possible features associated with said possible event and further in response to one or more of said possible events activating an exit associated with said possible event.

19. A method according to claim 18, wherein the event monitor comprises exits

25 associated with each of said possible events.

20. A method according to any of the preceding claims, wherein each document within the document databases contains information about its specific document type from a set of document types, the document type being significant for applicability with respect to the

30 access and use of the document, and each design element of the plurality of document databases for accessing and using documents also contains information about the specific document type which may be applied to it, the document type definition comprises the definition of a plurality of data fields with respect to the name and contents of each of the data fields, each document of a given document type comprises said plurality of data

35 fields of the definition.

21. A method according to claim 20, wherein at least one document type of the set of document types is defined commonly for the plurality of databases.

5   22. A method for performing an updating procedure on a plurality of document databases comprising design elements and comprised within one or several applications, of which design elements at least some comply with design features from a common library, the method comprising the steps of

including a new design feature in the common library for replacement of an

10   existing design feature of the common library from which the new design feature differs,

retrieving from at least one data storage structure associated with the common library information items about which document databases that comprise a design element which complies with the existing design, and

applying said new design feature to all of the databases that comprises a design

15   element which complies with the existing design feature according to said retrieved information items, except where replacing of the existing design element in the individual database is contradicted by other information.

23. A method according to claim 22 comprising a procedure of controlling a development

20   process of design elements or design features, the procedure comprising the steps of

selecting at least one design element or design feature,

marking the selected design element(s) or design feature(s) as user-reserved in a central data storage structure pertaining to said plurality of document databases,

producing (a) new design element(s) or design feature(s) to replace the selected

25   one(s),

undoing said marking of the selected design element(s) or design feature(s), and

replacing the previously selected design element(s) or design feature(s) with the respective new design element(s) or design feature(s),

the replacement of a design element or a design feature being prohibited if said design

30   element or design feature is marked as user-reserved in said central data storage structure.

24. A method according to claim 23, wherein the mark of the selected design element(s) or design feature(s) comprises data relating to a proprietor of the mark(s).

35

38

25. A method according to claim 24, wherein the step of undoing the marking only may be performed by the proprietor of the mark(s).

26. A method according to claim 24 or 25, wherein the step of replacing selected design
5   element(s) or design feature(s) with the respective new design element(s) or design feature(s) may be performed by the proprietor of the mark(s) even if the design element(s) or design feature(s) is (are) marked as user-reserved.

27. A method according to any of claims 23-26, further comprising the steps of
10   associating a label with a selected design element, a design feature or a document database and storing said label in a data storage structure, the label comprising sufficient data to restore the design element, design feature or document database with features in accordance with the features of the design element, design feature or document database at the time of association of the label,
15   restoring a design element, a design feature or a document database by use of data comprised within a label associated with said design element or design feature, and replacing the design element, design feature or document database with the restored design element, design feature or document database.

20   28. A method according to any of claims 22-27, wherein a technical foundation library within the common library comprises at least one design feature and each of the plurality of document databases comprises for each design feature of the technical foundation library a design element which complies with said design feature.

25   29. A method according to any of claims 22-28, further comprising storing, in the at least one data storage structure, information items relating to which document databases the new design feature has been applied to.

30. A method according to any claims 22-29 comprising at least one update procedure
30   constituting a maintenance step performed on the document databases, each update procedure comprising the steps of
           including a new design feature in the common library,
           applying said new design feature to all or some of the document databases, and
           storing, in said at least one data storage structure, information items relating to
35   which document databases the design feature has been applied to.

31. A method according to any of claims 22-30, comprising at least one include procedure for including a previously generated document database into the plurality of document databases maintained from the common library, the previously generated document

5   database comprising at least one design element which complies with a design feature of the common library, the include procedure comprising the step of storing information items about which design features from the common library design elements of the document database comply with and saving the information item(s) in said at least one data storage structure.

10

32. A method according to any of claim 22-31, wherein protection information items relating to at least some of the design features of the plurality of document database that comply with a design feature of the common library is stored, each of the stored protection information items regarding whether replacement of a specific design element in a specific

15 document database in a subsequent update procedure performed on the databases is admissible, said protection information items being used by the at least one update procedure to determine whether replacement of existing design features in individual document databases is contradicted.

20 33. A method according to claim 32, wherein the protection information items are stored in the at least one data storage structure associated with the common library.

34. A method according to claim 32, wherein the protection information items are stored within the individual document databases.

25

35. A method according to any of claims 22-34, wherein the document databases are generated according to the standards of Lotus Notes or a modification or further development thereof.

30 36. A method according to any of claims 22-35, wherein the operation of a plurality of said design elements complying with a design feature from the common library comprises the steps of
        activating an exit of the design element,

retrieving a possible feature being associated with the specific exit from at least at least one setup definition defining features of the individual document database to be applied to design elements of the individual document database, and

applying said possible feature.

37. A method according to claim 36, wherein at least one of said plurality of design elements is an event monitor being associated with and applied by at least one of the design elements of the document database complying with a design feature of the common library, the operation of the event monitor comprising the step of monitoring each possible event associated with the design element by which it is applied, and in response to the occurrence of one or more of said possible events the steps of applying one or more possible features associated with said possible event and further in response to one or more of said possible events activating an exit associated with said possible event.

38. A method according to claim 37, wherein the event monitor comprises exits associated with each of said possible events.

39. A method according to any of claims 22-38, wherein each document within the document databases contains information about its specific document type from a set of document types, the document type being significant for applicability with respect to the access and use of the document, and each design element of the plurality of document databases for accessing and using documents also contains information about the specific document type which may be applied to it, the document type definition comprises the definition of a plurality of data fields with respect to the name and contents of each of the data fields, each document of a given document type comprises said plurality of data fields of the definition.

40. A method according to claim 39, wherein at least one document type of the set of document types is defined commonly for the plurality of databases.

41. A system comprising a plurality of document databases comprised within one or more applications and residing on data storage means of a computer system comprising one or more computers, each of the document databases comprising at least some design elements which comply with design features from a common library, the system comprising at least one data storage structure containing, for each of the design features

of the common library, information about which of the plurality of document databases that comprise a design element which complies with the design feature.

42. A system according to claim 41, wherein one of the applications is a control
5   application for controlling a development process of design elements or design features, the control application being adapted for performing a procedure comprising the steps of
        selecting at least one design element or design feature,
        marking the selected design element(s) or design feature(s) as user-reserved in a central data storage structure pertaining to said plurality of document databases,
10   producing (a) new design element(s) or design feature(s) to replace the selected one(s),
        undoing said marking of the selected design element(s) or design feature(s), and
        replacing the previously selected design element(s) or design feature(s) with the respective new design element(s) or design feature(s),
15   the replacement of a design element or a design feature being prohibited if said design element or design feature is marked as user-reserved in said central data storage structure.

43. A system according to claim 42, wherein the mark of the selected design element(s)
20   or design feature(s) comprises data relating to a proprietor of the mark(s).

44. A system according to claim 43, wherein the step of undoing the marking only may be performed by the proprietor of the mark(s).

25 45. A system according to claim 43 or 44, wherein the step of replacing selected design element(s) or design feature(s) with the respective new design element(s) or design feature(s) may be performed by the proprietor of the mark(s) even if the design element(s) or design feature(s) is (are) marked as user-reserved.

30 46. A system according to any of claims 42-45, wherein the procedure further comprises the steps of
        associating a label with a selected design element, a design feature or a document database and storing said label in a data storage structure, the label comprising sufficient data to restore the design element, design feature or document database with features in

accordance with the features of the design element, design feature or document database at the time of association of the label,

      restoring a design element, a design feature or a document database by use of data comprised within a label associated with said design element or design feature, and

5   replacing the design element, design feature or document database with the restored design element, design feature or document database.

47. A system according to any of claims 41-46, wherein a technical foundation library within the common library comprises at least one design feature and each of the plurality

10 of document databases comprises for each design feature of the technical foundation library a design element which complies with said design feature.

48. A system according to any of claims 41-47, wherein the at least one data storage structure resides on storage means of a computer system which is different from the

15 computer system on which the plurality of document databases resides.

49. A system according to any of claim 41-48, wherein the at least one data storage structure resides on the computer system on which the plurality of document databases resides.

20

50. A system according to any of claims 41-49, wherein the at least one data storage structure is associated with the common library.

51. A system according to any of claims 41-50, wherein one of the applications is an

25 update application for updating design features of the common library, the update application being adapted for performing an update procedure comprising the steps of

      including a new design feature in the common library for replacement of an existing design features of the common library from which the new design feature differs,

      retrieving information items from the at least one data storage structure relating to

30 which document databases that comprise a design element which complies with the existing design, and

      applying said new design feature to all of the databases that comprises a design element which complies with the existing design feature according to said retrieved information items, except where replacing of the existing design element in the individual

35 database is contradicted by other information.

52. A system to claim 51, wherein the update application is further adapted for storing, in the at least one data storage structure, information items relating to which document databases the new design feature has been applied to.

5

53. A system according to any of claims 41-52, wherein one of the applications is an update application for updating the common library by including a new design feature, the update application being adapted for performing an update procedure comprising the steps of

10   including a new design feature in the common library,
          applying said new design feature to all or some of the document databases, and
          storing, in said at least one data storage structure, information items relating to which document databases the design feature has been applied to.

15 54. A system according to any of claims 41-53, wherein one of the applications is an include application for including a previously generated document database into the plurality of document databases maintained from the common library, the previously generated document database comprising at least one design element to which at least one of the design features of the common library has been applied, the include application
20 being adapted for performing an include procedure comprising the step of storing information about which design features have been applied to said document database and saving the information item(s) in said at least one data storage structure.

55. A system according to any of claims 51-54, wherein protection information items
25 relating to at least some of the design features of the plurality of document database that comply with a design feature of the common library is stored, each of the stored protection information items regarding whether replacement of a specific design element in a specific document database in a subsequent update procedure performed on the databases is admissible, said protection information items being used by the at least one update
30 procedure to determine whether replacement of existing design features in individual document databases is contradicted.

56. A system according to claim 55, wherein the protection information items are stored in the at least one data storage structure associated with the common library.

35

44

57. A system according to claim 55, wherein the protection information items are stored within the individual document databases.

58. A system according to any of claims 41-57, wherein the databases are generated
5   according to the standards of Lotus Notes or a modification or further development thereof.

59. A system according to any of claims 41-58, wherein each of said document databases comprises at least one setup definition defining features of the individual document
10 database to be applied to design elements of the individual document database, said design elements complying with a design feature of the common library, a plurality of said design elements each comprising one or more exits at which the design element is adapted for retrieving from the at least one setup definition a possible feature associated with the specific exit and applying said possible feature.
15

60. A system according to claim 59, wherein at least one of said plurality of design elements is an event monitor associated with and applied by at least one of the others of said design elements complying with a design feature of the common library, the event monitor being adapted to monitor each possible event associated with the design element
20 by which it is applied and to apply features according to one or more of said possible events, the event monitor comprising exits for at least some of said possible events.

61. A system according to claim 60, wherein the event monitor comprises exits for each of said possible events.
25

62. A system according to any of claims 41-61, wherein each document within the document databases contains information about its specific document type from a set of document types, the document type being significant for applicability with respect to the access and use of the document, and each design element of the plurality of document
30 databases for accessing and using documents also contains information about the specific document type which may be applied to it, the document type definition comprises the definition of a plurality of data fields with respect to the name and contents of each of the data fields, each document of a given document type comprises said plurality of data fields of the definition.
35

45

63. A system according to claim 62, wherein at least one document type of the set of document types is defined commonly for the plurality of databases.

64. A method of controlling a development process of a plurality of document databases
5   comprising design elements and comprised within one or several applications, the method comprising the steps of

selecting at least one design element,

marking the selected design element(s) as user-reserved in a central data storage structure pertaining to said plurality of document databases,

10   producing (a) new design element(s) to replace the selected one(s),

undoing said marking of the selected design element(s), and

replacing the previously selected design element(s) with the respective new design element(s),

the replacement of a design element of the plurality of document databases being

15   prohibited if said design element is marked as user-reserved in said central data storage structure.

65. A method according to claim 64, wherein the mark of the selected design element(s) comprises data relating to a proprietor of the mark(s).

20

66. A method according to claim 65, wherein the step of undoing the marking only may be performed by the proprietor of the mark(s).

67. A method according to claim 65 or 66, wherein the step of replacing selected design
25   element(s) with the respective new design element(s) may be performed by the proprietor of the mark(s) even if the design element(s) is (are) marked as user-reserved.

68. A method according to any of claims 64-67, further comprising the steps of

associating a label with a selected design element or a document database and

30   storing said label in a data storage structure, the label comprising sufficient data to restore the design element or the document database with features in accordance with the features of the design element or document database at the time of association of the label,

restoring a design element or document database by use of data comprised within

35   a label associated with said design element, and

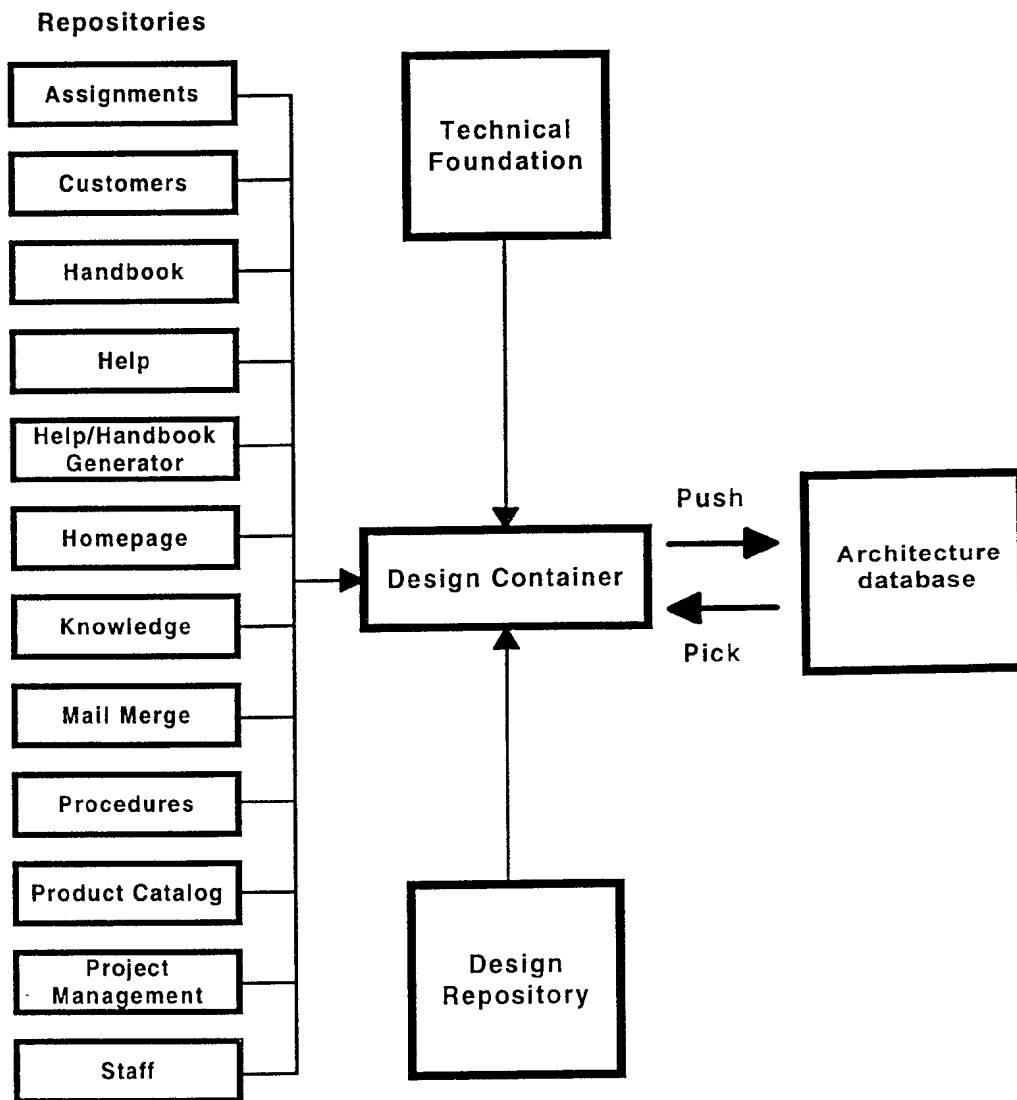replacing the design element or document database with the restored design element or document database.

5

Repositories
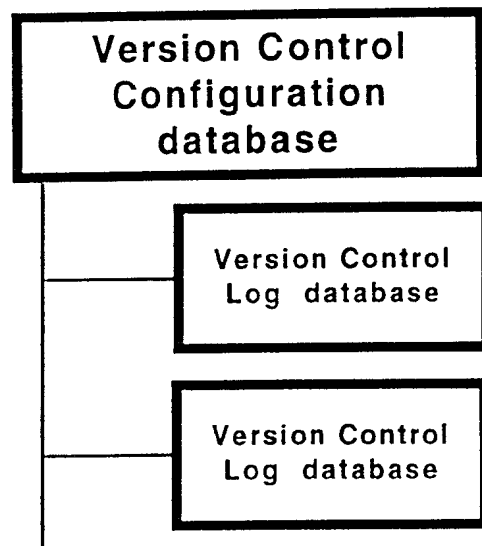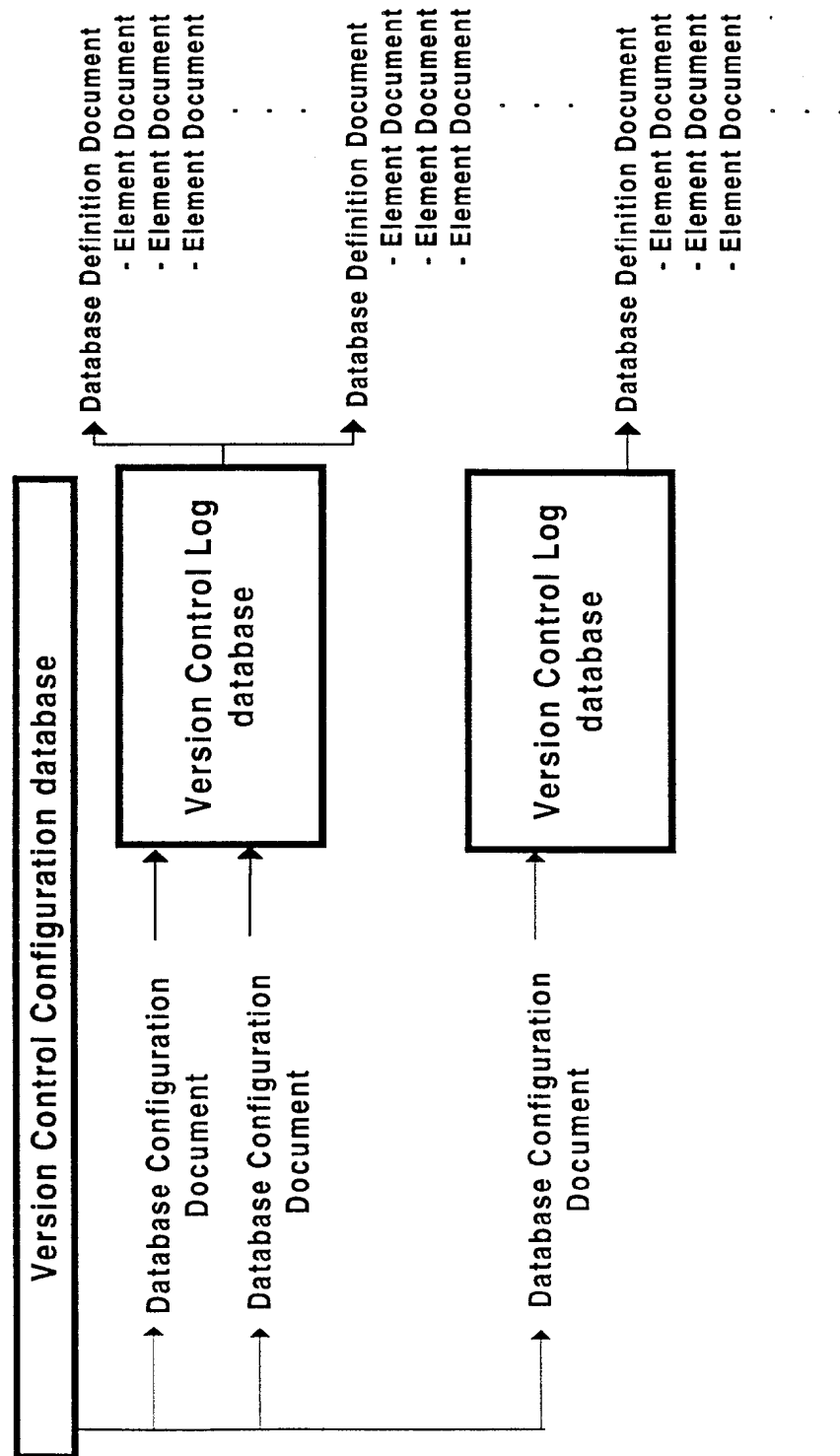

Fig. 1

```
┌─────────────────────────────┐
│     Version Control         │
│     Configuration           │
│     database                │
└─────────────────────────────┘
       │
       │      ┌──────────────────────┐
       │      │   Version Control    │
       ├──────│   Log  database      │
       │      └──────────────────────┘
       │
       │      ┌──────────────────────┐
       │      │   Version Control    │
       ├──────│   Log  database      │
       │      └──────────────────────┘
       │
```

Fig. 2

Fig. 3